Naval Research Laboratory

Washington, DC 20375-5320

NRL/FR/5522--11-10,201

Distributed Service Discovery within Mobile Ad Hoc Networks

Joseph P. Macker
Justin W. Dean
Ronald D. Lee
Robert B. Adamson

Protean Research Group
Information Technology Division

Ian Taylor
Andrew Harrison

Computer Science Department
Cardiff University
Wales, UK

September 20, 2011

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)* 20-09-2011	**2. REPORT TYPE** Formal Report	**3. DATES COVERED** *(From - To)* 2003-2009

4. TITLE AND SUBTITLE Distributed Service Discovery within Mobile Ad Hoc Networks	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER 0602235N
6. AUTHOR(S) Joseph P. Macker, Justin W. Dean, Ronald D. Lee, Robert B. Adamson, Ian Taylor,* and Andrew Harrison*	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER 55-9638-C15
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Research Laboratory, 4555 Overlook Ave., SW, Washington, DC 20375-5320	**8. PERFORMING ORGANIZATION REPORT NUMBER** NRL/FR/5522--11--10,201
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Office of Naval Research, 875 N. Randolph St., Arlington, VA 22203; Cynthia Dion-Schwarz, Ph.D.,SES, Director, Information Systems & Cyber Security, OASD(R&E)/Research, 1777 N. Kent Street, Rosslyn, VA 22209	**10. SPONSOR / MONITOR'S ACRONYM(S)** ONR and OSD/DDR&E/NCCP
	11. SPONSOR / MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

* Cardiff University, Wales, UK

14. ABSTRACT

The potential for operating more mobile and dynamic networking infrastructures is raising new challenges in developing effective service registration and discovery technologies. Discovering and maintaining services within dynamic wireless networks is a challenge requiring novel mechanisms for more effective service detection, tracking, and adaptation. We discuss early prototyping of a service discovery framework based upon the service location protocol (SLP) called extended SLP (xSLP). xSLP has been applied within mobile ad hoc network (MANET) environments to investigate design tradeoffs in both service discovery and registration. Simulations have been performed to examine three multicast routing algorithms and unicast routing in combination with xSLP. A fundamental challenge is operating with varying bandwidth, topological dynamics, errored links, and dynamic failure events. A fundamental goal of the work is to design more effective service discovery mechanisms for use in these types of challenged wireless network environments and to understand the protocol interactions at the routing and reliable transport layer.

15. SUBJECT TERMS

Service discovery	xSLP	Wireless networking	Routing algorithms	Network simulation
Mobile Ad Hoc network	MANET	Multicast	Service registration	

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Justin W. Dean
a. REPORT Unclassified	**b. ABSTRACT** Unclassified	**c. THIS PAGE** Unclassified	UL	16	**19b. TELEPHONE NUMBER** *(include area code)* (202) 767-3397

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

CONTENTS

DISTRIBUTED SERVICE DISCOVERY WITHIN MOBILE AD HOC NETWORKS

INTRODUCTION

The potential for operating more mobile and dynamic networking infrastructures is raising new challenges in developing effective service registration and discovery technologies. Discovering and maintaining services within dynamic wireless networks is a challenge requiring novel mechanisms for more effective service detection, tracking, and adaptation. We discuss early prototyping of a service discovery framework based upon the service location protocol (SLP) called extended SLP (xSLP). xSLP can be viewed as a set of configuration enhancements of the existing core SLP framework that can applied more generally to a variety of existing service discovery (SD) frameworks. We are interested in examining service discovery design and deployment tradeoffs within mobile ad hoc network (MANET) environments. In this report, we describe an initial experimental use of xSLP within a mobile topology and we explore a number of mobile ad hoc multicast approaches to assist service discovery within dynamic multiple hop networks. Simulations have been performed to examine three multicast routing algorithm variants and unicast routing in combination with xSLP. A fundamental goal of the work is to evaluate and design more effective service discovery mechanisms for use in these types of challenged wireless network environments and to understand the protocol interactions at the routing and reliable transport layer.

BACKGROUND AND MOTIVATION

Robustness, flexibility, and a minimum need for user intervention are keys to a well architected and deployable MANET solution [1]. While much work has been done developing routing solutions for MANET, development of service-based mechanisms applicable to MANET environments has not received the same attention. These mechanisms would allow devices to publish and locate network services along with related attributes and without these service discovery solutions network utility will be reduced.

Industry and standard consortiums have been developing approaches for some time now relating to network service discovery. These service discovery protocols (SDPs) are increasingly becoming an essential part of middleware and evolving service-based network architectures (e.g., Service Oriented Architecture (SOA)). There are a myriad of service discovery approaches with varying characteristics, maturity, and purposes; a few examples include Sun's JINI [2], IETF's SLP [3], Microsoft's UPnP [4], Bluetooth's SDP [5], and Apple's Bonjour [6]. Most of these approaches have significant limitations or immaturity of understanding when applied to distributed, wireless ad hoc environments. Yet, this is exactly the type of environment that could best benefit from adaptive, robust service discovery.

Manuscript approved July 8, 2010

Current motivational application scenarios range from running collaborative fault tolerant multimedia sessions across hybrid infrastructure-MANET networks to exposing and interacting with enterprise level services transparently across differing network environments. In these heterogeneous network scenarios, broker services operating at the edges of these networks are needed to provide translational and publish/subscribe services. These brokers form a common communication bridge between networks. Current wide area network standards for discovery in both of these areas are not satisfactory for supporting MANET networks since they are mostly based on TCP-centric services, centralized lookup services, or both. Our alternative approach provides a more flexible and dynamic set of building blocks that can be used to build appropriate service discovery mechanisms for the envisioned deployment environment. In this regard, one key criterion for our envisioned service discovery needs is the ability to configure the system to work in a variety of network environments, ranging from highly dynamic time-varying networks (e.g., MANET) to the more stable infrastructure networks.

PROBLEM FOCUS AND DEFINITION

A service may be any hardware or software feature that can be used by an end user or process. For the discovery of services, typically two core mechanisms are used: multicast discovery and unicast lookups. Multicast messaging can be used for client-side, reactive-based discovery (dynamic discovery), and server-side, proactive-based announcement of services. Currently, these messages are often limited to one hop operation due to the use of link local messaging. Unicast messaging can be used for connection to specific known service directories that contain look up tables for other services that are available on the network. In general, service directories are used in centralized or federated cases. They are typically discovered first and then contacted directly to search for the desired service or services. Other protocols (e.g., anycast routing providing a one-to-any delivery) have also be investigated but they are less common in current middleware stacks.

In most of today's distributed applications and middleware, some concept of a service directory capability is used. For example, for a service directory: in Jini, it is the LUS (lookup server); in a super-peer network it is a super-peer; in Napster [7], it is the central Napster server; in JXTA [8], it is a rendezvous; in Web services, UDDI [9] may be used as the directory service; and in service discovery protocols, such as SLP, it is called a directory agent or DA.

A combination of multicast and unicast techniques is sometimes supported by SDP frameworks. In these cases, an entity could use multicast to auto-detect the location of a service directory address on the network then use a unicast connection to connect to the discovered service directory address. This is of limited use in practice, however, due to the multicast messaging operating with link local scope. This limits discovery of service directories to single-hop neighbors. Furthermore, these multicast techniques do not currently support multiple-hop network operations and are not easily modified to support such operation.

A number of systems, such as Jxta, Bonjur, Zeroconf, Jini, and SLP, support some form of multicast. An application may even choose to opt out of using a service directory, and simply use multicast to auto-configure connections between clients and servers directly to form a more direct peer-to-peer (P2P) service lookup and connection. This P2P approach is the focus of our initial experiments described in this report. Further work is under way to examine brokering, caching, and proactive notification/presence.

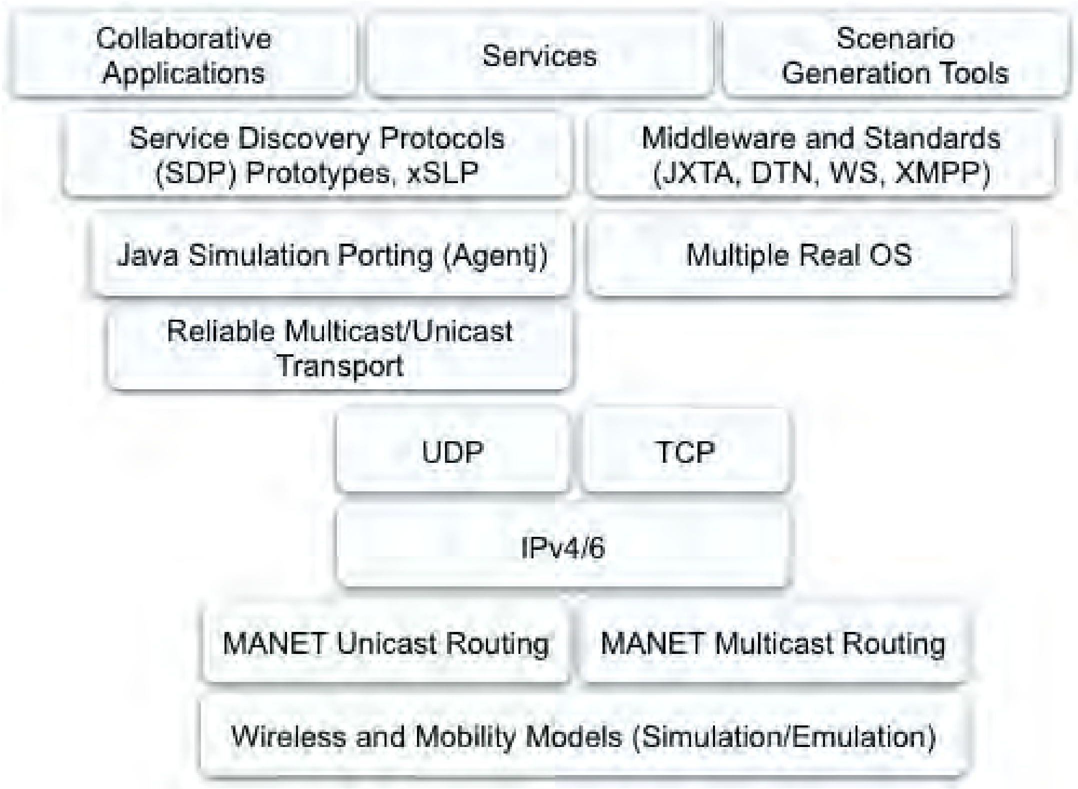
Collaborative Applications
Services
Scenario Generation Tools
Service Discovery Protocols (SDP) Prototypes, xSLP
Middleware and Standards (JXTA, DTN, WS, XMPP)
Java Simulation Porting (Agentj)
Multiple Real OS
Reliable Multicast/Unicast Transport
UDP
TCP
IPv4/6
MANET Unicast Routing
MANET Multicast Routing
Wireless and Mobility Models (Simulation/Emulation)

Fig. 1 – Overview of the protocols and standards being investigated for bridging MANET and infrastructure networks

Figure 1 shows the overview of various software and protocol components in our network, simulation, emulation, and services stack. The collaborative applications envisioned (e.g., chat and Situational Awareness) can use high-level XML-based standards for communication based on XMPP [10] and Web Services (WS) standards including WS-Eventing. Alternatively, the applications can choose some other means of data modeling. Our design and use of Extended Service Location Protocol (xSLP) is indifferent to specific application layers, data models, or other middleware transport techniques. These applications can be fed into scenario generation tools to create either emulations or simulations of the software. The applications can use a combination of TCP, UDP, and multicast data messaging. Different types of MANET unicast/multicast routing protocols and algorithms can be used independent of other protocol components. When using ns-2 simulations or emulation, the scenarios use real code, i.e., they do not approximate network activity by using a set of discrete events. In the case of ns-2 simulation, Java applications are not aware that they are being run in simulation. Therefore, the results presented in this report show accurate representations of how this software would perform on a real network with the same properties as the simulated one. We believe this approach is rigorous and stress-tests the software to its limits. The resultant software is more thoroughly tested and debugged, leading to deployment-ready applications and middleware stacks.

In this report, we investigate the combination of these layered techniques for service discovery in MANET. First, we outline an initial set of experiments that apply our xSLP framework to control a number of independent variables at the SDP layer. xSLP supports the notion of a reliable convergence method using multiple retries, and we allow this algorithm to be parameterized from zero to n retransmission attempts (retries) in order to examine this effect on reliability and delay independent from routing and other system variables. xSLP is further tested in conjunction with both mobile multicast and unicast routing protocols, thus enabling distributed service discovery within several mobile ad hoc network scenarios. We use xSLP in its active multicast querying mode and explore the use of Simple Multicast Forwarding (SMF) [11] variants for underlying MANET multicast support of xSLP queries. We also compare the tradeoffs involved in applying both multicast and unicast responses to such multicast querying scenarios. We then compare their relative performance in terms of discovery success and delay as dynamics increase.

Our initial findings suggest that related retransmission convergence algorithms have a significant impact on increasing the success rate of discovery in all cases. Our results also suggest that differing multicast variants provide significantly different results when examined across a variety of mobile scenarios. Results through simulation are presented that run an actual enhanced Java-based SLP implementation within the DARPA ns-2 simulation environment through the use of AgentJ [12]. AgentJ provides a bridge between the Java code and the C++ ns-2 protocols and allows native Java-developed code to operate as is within the ns-2 environment.

PREVIOUS WORK

Although related research in this area is limited, past work has been performed in examining service discovery designs and tradeoffs within MANET environments. Sailhan and Issarny [13] presented a design and analysis of a scalable MANET service discovery approach for larger MANETs. Some of their work focuses on using Web Service Discovery Language (WSDL) based service templates. In general, we are not convinced this is a good idea in a MANET because of the duplication of information in WSDL and its associated overhead. They also largely concentrate on the problem of using distributed directories to improve performance. We believe this is an important contribution. However, they provide only limited evidence in demonstrating that the use of their decentralized directory approach outperforms a purely pull-based reactive flooding approach in terms of overhead produced as the periodicity of querying increases. They also provided only mild mobility for studied scenarios. We believe our results provide novel findings that should be incorporated in revisiting these tradeoff studies, especially given the potential for more significant mobility, improved connected dominating set (CDS) flooding, and other system dynamics. Kozar and Tassiulas [14] tightly couple their service discovery to a dynamic multicast backbone that is formed and maintained within the MANET. Their work is closely related to ours; however, we further decouple the multicast backbone from the service discovery framework. We also focus on examining multiple multicast approaches with and without unicast responses. Our SDP and routing frameworks are based upon rich existing, deployable specifications and non-abstracted designs within the simulator. We believe heterogeneous networks also require multiple approaches, and we focus on gatewaying our SDP to interoperate with other heterogeneous systems rather than solving the problem of large-scale flat operation. The Konark protocol [15], also often cited, provides a highly distributed SDP approach by using a mini-HTTP server at each node and by using Simple Object Access Protocol (SOAP) messaging standards. Konark is similar to our work in that it supports both push and pull mechanisms for service discovery. Early Konark work mostly concentrates on the design of the service templates and the maintenance of the registry data structures. More recently, they have been examining a tightly integrated service discovery and routing approach. Our approach is more decoupled and provides additional advantage by supporting and examining more efficient flooding mechanisms. In other words, we provide novel results and work in the area of examining the interactions of underlying protocols and SDP mechanisms within dynamic network environments without forcing a design coupling. Our results and flexible framework can be applied more generally to any design, including the Konark work.

SLP AND XSLP

The Service Location Protocol [3] allows services to be deployed and discovered with minimal configuration. It was a design predecessor of widely deployed protocols (such as Apple Bonjour and ZeroConf) and is one of the several standardized protocols for service discovery. It employs the combined use of unicast, multicast and distributed directory agent techniques to achieve dynamic deployment and discovery capabilities.

SLP breaks the deployment and discovery process into three roles: User Agent (UA), a service consumer; Service Agent (SA), a service provider; and Directory Agent (DA), a cache for service advertisements. Each SLP agent produces and consumes certain types of messages, which can also be scoped so that messages can be filtered. Services can also expose attributes that are key/value pairs. These can be queried by a UA using a filter expression. The combination of scopes, attributes, and service types provides a simple yet flexible means of describing and querying for services.

When using multicast to retrieve service adverts, SLP uses a convergence algorithm to counteract possible packet loss. Specifically, a request is repeated if no response is received, along with the use of an appropriate backoff algorithm. Typically, if no responses are received after two attempts, the request is dropped. When sending responses, both SAs and DAs include their own address in the message. This allows UAs to know whether they have already received a response from the particular node before.

xSLP includes a set of enhancements we have developed for the configuration of SLP, allowing us to apply it flexibly to multiple modes of operation within multiple architectural configurations. In particular, we have been looking into implementation-level details that can help tune the algorithm to the environment at hand, such as being able to set the Xid (service identifier) caching period for dealing with stale service advertisements. Further, we have implemented a distributed opportunistic caching scheme, which is based on the idea that if a client queries for a service then perhaps other clients can opportunistically cache responses or adverts, especially when multicast is used for the service transmission. Lastly, we have implemented some fine-tuning mechanisms that allow us to change default SLP parameters. For example, in our experiments, we allow fine tuning of the convergence retry algorithm to allow for more retries than the default of 2, which allow us to run up to four retries in the experiments in this report. The histograms in the experimental results section show the distributions of the retries and that retries of more than two are often required for high robustness in the dynamic wireless networks studied.

EXPERIMENTAL APPROACH AND ISSUES

In this section, we describe the simulation environment and parameters used for various simulation runs. For simulation of SLP, we use Agentj [12], which is a tool developed in the group that allows the execution of unmodified Java applications within the ns-2 simulation environment. It uses a combination of factory socket implementations, Java bytecode rewriting, aspect-oriented techniques, and Java instrumentation to modify an application's configuration (and class files) at run time to use Agentj's replacement classes for the implementation within ns-2. Agentj contains a complete socket implementation (UDP, multicast, and TCP), a DNS implementation (for converting between IPv4/6 addresses and ns-2 addresses), a modified thread implementation (for synchronizing multiple application threads within the single threaded nature of ns-2), replaced thread synchronization mechanisms (for both java.util.concurrent and java.lang.Object's wait and notify), and reimplementations of timing methods from real-time to ns-2 time (currentTimeMillies and java.util.Timer).

The underlying implementations of Agentj's Java classes use the C++ Protolib [16] toolkit (accessed though the Java Native Interface (JNI)), which has bindings for ns-2 sockets, timers, event handling, and other operating specific system functions. Agentj also provides bootstrapping for the application steps of Java objects using a familiar command style interface. For the studies described here, we used unmodified xSLP code, developed in native Java, to run the simulations over a MANET network within ns-2. Agentj, therefore, runs real code rather than approximating code through simulation of the steps extracted from the logic of the code, and as a result provides an accurate representation of the SLP algorithm and how it would work in a real network. This not only enables more accurate simulations but also provides an excellent debugging environment for distributed network communications.

Simulations, Mobility, and Parameters

The ns-2 nodes were configured with a communication range of 250 m using the 802.11 wireless ns-2 model. For each data point, we averaged over 10 distinct simulations of 10 minute duration. SLP event generation and motion paths were predetermined; SLP events were generated with a Poisson distribution with an average interval of 6 seconds per client. Motion was restricted to a 300 m by 1500 m grid, as in Konark [15], and generated using a random waypoint model with speeds ranging from 1 to 20 m/s and 0 pause time for each of the 50 nodes. Network analysis was performed on all generated motion files to guarantee full connectivity throughout each run, assuming instantaneous network convergence were possible. This was to avoid fragmentation events from artificially skewing results. Future work will examine actual topological fragmentation scenarios.

SMF is the fundamental multicast forwarding method applied in our studies along with variants for CDS optimization. Classical flooding was used as a baseline with ECDS [17] and S-MPR [18] being the two other relay set optimization methods used. Simulations were run with 50 nodes, 10 clients, and one to three servers to illustrate any trends due to differing network deployments. Maximum nodal speeds were varied between 1, 2, 4, 8, 12 and 20 to demonstrate network mobility convergence trends between the various discovery methods simulated. While request delivery was always performed using multicast, simulations were performed using both multicast and unicast methods for server to client replies. Where unicast communication was used, NRLOLSR [19], an NRL developed OLSR implementation, provided unicast network functionality. Multicast replies used the same method as discovery in a given simulation. We also tested the xSLP reliability mechanism, which used up to four request retransmissions in conjunction with exponential back-off timers.

Performance Metrics

The main performance metric we were interested in measuring was the discovery success ratio, simply the number of successful discovery events divided by the number of unique discoveries attempted. Application level retransmissions due to the reliability mechanism were not counted as unique discovery attempts. Service request delivery ratio was also measured by dividing the number of uniquely delivered service requests aggregated across all servers divided by the number of unique service requests amongst all clients. This measurement allows a distinction to be made between request delivery and request reply ratios. In addition to success rate, the delay required per successful request-reply combination was also measured. Network overhead induced by the service discovery was measured independently of routing overhead required to provide the network functionality of a discovery method. In many networks, both multicast and unicast functionality will be required, negating any possible saved overhead from turning off unicast routing.

EXPERIMENTAL RESULTS

Figure 2 shows the average successful SLP request rate with differing underlying multicast routing algorithms and varying SLPs reply delivery methods: unicast or multicast. ECDS using multicast replies is the top performer. Using multicast replies, instead of unicast, increased success rates when ECDS was used but decreased success rates when Classical and SMPR were enabled. The reason for this decrease differed, however, with Classical flooding causing congestion and SMPR flooding providing no better routing performance due to the underlying unicast routing protocol, OLSR, being based on SMPR link state information.

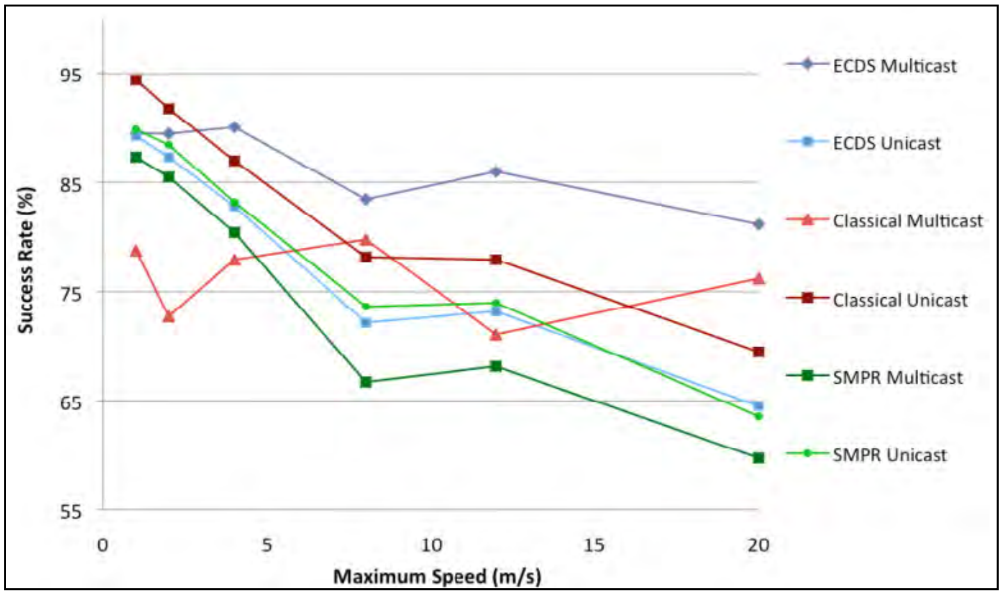

Fig. 2 – A comparison of the different SMF routing algorithms operating with nodes moving at six different speeds and with both unicast and multicast responses

Figure 3 demonstrates the resultant delay measurements when using multicast replies. SMPR delays were consistently lower than both ECDS and Classical due to SMPR flooding guaranteeing shortest hop flooding paths while not reaching network congestion points. Figure 4 plots atomic request success events for a sample run using ECDS flooding unicast replies.

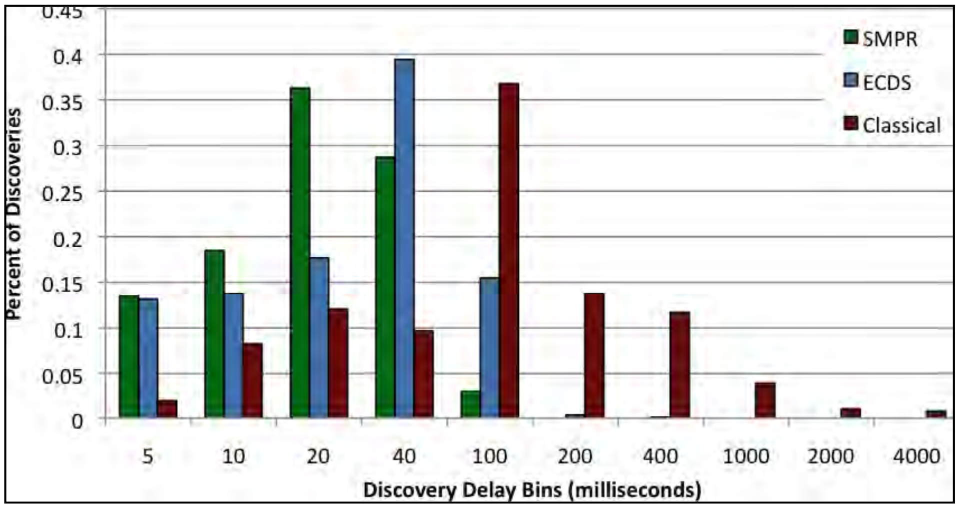

Fig. 3 – A histogram showing the delay times for discovery across all query events for classical flooding and ECDS routing

While network mobility was generated to guarantee connectivity, the physical location of the sole server was such that ECDS could not reach convergence with restricted connectivity and high levels of localized traffic. More than one server or intelligent placement of server nodes would prevent this type of behavior, as is demonstrated in Fig. 6.

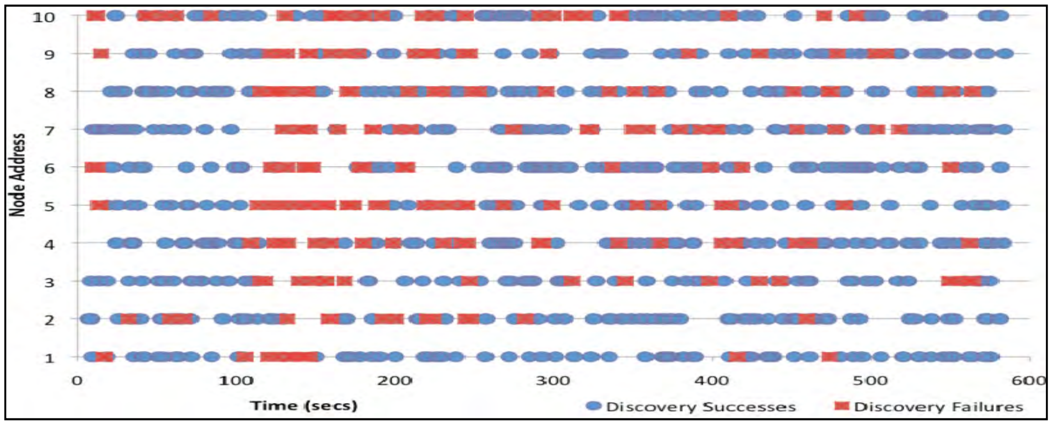

Fig. 4 – The success and failures from all ten-client nodes during the course of the simulation

SDP protocols can provide some method of reliability. Figure 5 demonstrates successful discovery rates of the same simulation runs as Fig. 2, but with a simple exponential timer back-off method implemented within SLP to retry service requests. Success rates improved across the board with both Classical and ECDS using multicast replies maintaining an average of approximately 99% across all runs, which is an average of a 22% improvement over Fig. 2.

Figure 6 compares discovery success rates in the same set of simulation runs using ECDS routing with multicast replies and SLP retries. However, we show the effect on increasing the number of SLP servers into the network and therefore increasing availability in order to counter the effects observed in Fig. 4 above. Additional improvement is achieved with the introduction of more servers, with average rates for the two-server case of 99.2% and for the three-server case of 99.4%, across all mobility scenarios. Considering the severity of the latter mobility patterns, these results achieve a sustained level of robust reliability.

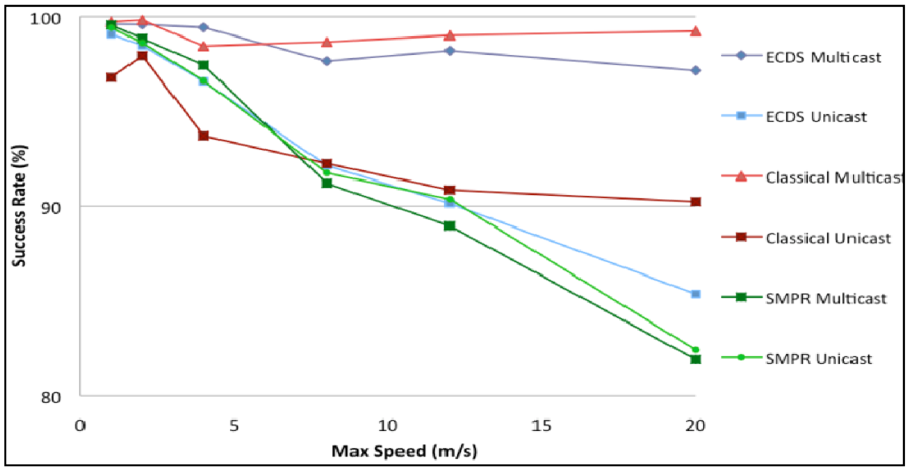

Fig. 5 – The different SMF routing algorithms coupled with SLP convergence algorithm

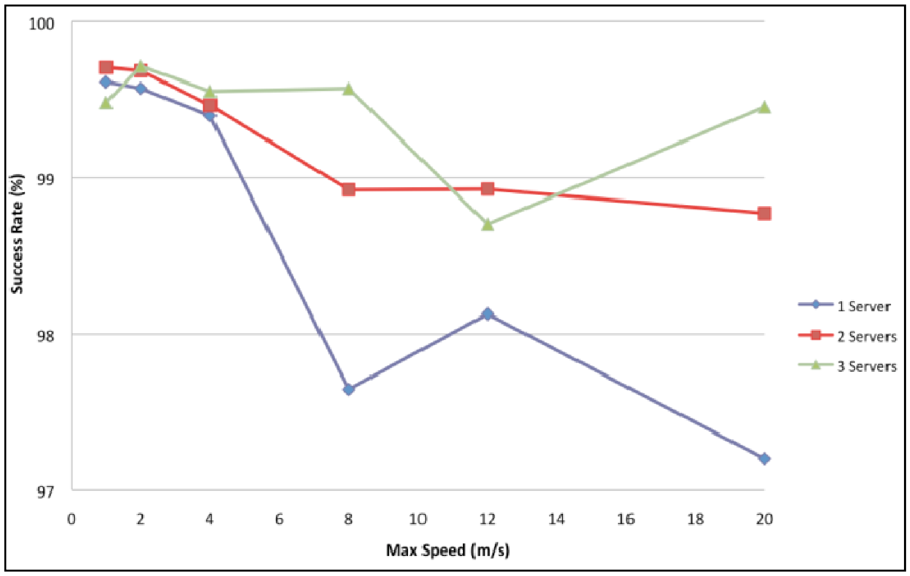

Fig. 6 – ECDS routing is shown with different numbers of servers to increase availability on the network

The reliability of the success rates achieved in Fig. 6 demonstrates the adaptability of the SLP retries when used in conjunction with the underlying multicast routing protocol. Figures 7 and 8 show histograms of SLP retries across all mobility scenarios for the one-server and three-server cases respectively.

For the three-server case in Fig. 8 we can see that fewer retries are needed than for the one-server case (Fig. 7) because there is more server availability in the network. In these cases, the average number of retries required for the one-server case was 28% of the total successful queries, while for the three-server case, just 13% of queries required retries, showing that increasing the availability also has a significant impact on the success rates achieved overall. Further, for these cases, the effect of adding servers reduces the overall discovery delay rates. For the one-server case, delays were averaging 169 ms, while for the three-server case, they averaged 97 ms (an improvement of approximately 40%).

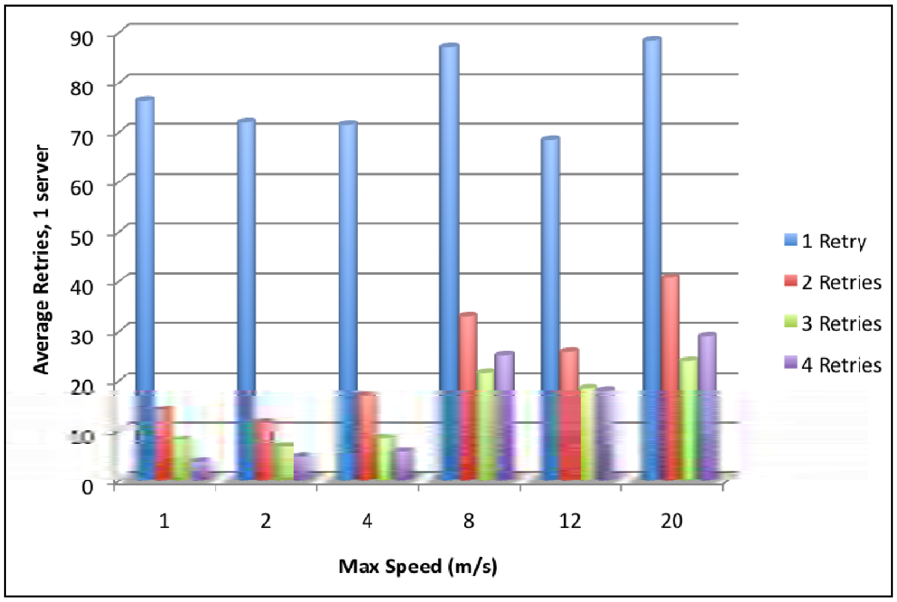

Fig. 7 – A histogram of retries is shown for the one-server, ten-client case, using ECDS routing in SMF

Further, it can be seen in Fig. 8 that as the underlying protocol begins to fail for the more severe mobile cases, the SLP convergence algorithm maintains successful rates through issuing more retry queries. For example, for the 20 m/s case, the number of retries grows higher as previous retries lead to failure. Ultimately statistics prevail, and upon the second, third, or fourth retry, success is achieved in most cases.

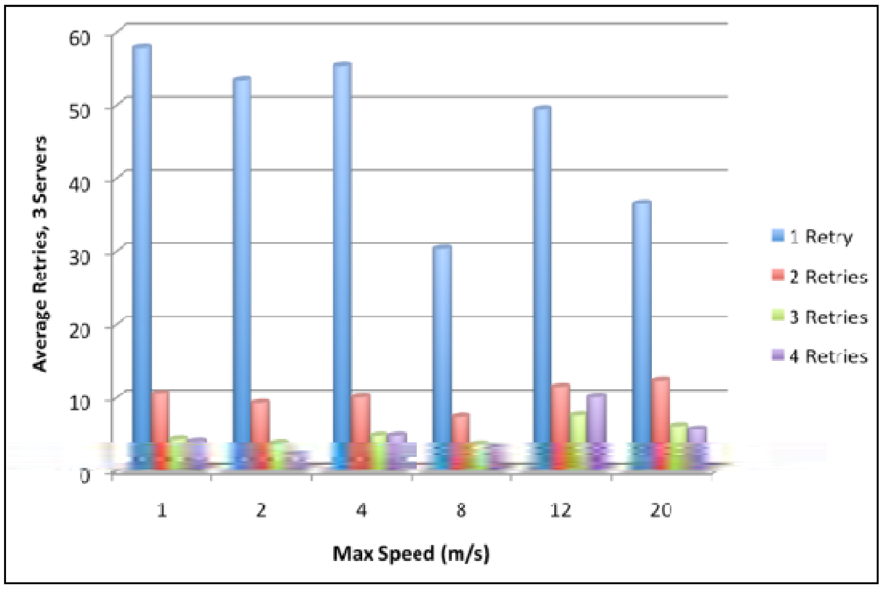

Fig. 8 – A histogram of retries is shown for the three-server ten-client case, using ECDS routing in SMF

CONCLUSION

The scenarios discussed in this report were tested using a real Java SLP implementation running within the ns-2 simulation environment, where timers, protocols, and threads were interacting as they would in a real-world setting. In our tests, we conclude with one qualitative finding and three quantitative findings.

Qualitatively, the xSLP code was run within ns-2 without modification using the AgentJ system. This approach, we believe, will have an enormous impact on the success of deployment code since it will be more thoroughly tested and improved in scaled and challenging scenarios.

First, our quantitative results indicate that ECDS-based SMF multicast forwarding proved to have the most attractive characteristics overall over Classical and SMPR, achieving a combination of high success rates with low latency discovery times for scenarios tested. Second, by using our parameterization of SLP retry convergence methods, we were able to examine query retransmission in a decoupled manner from routing delivery and showed significant improvements across the board for all three multicast protocol variants and for both unicast and multicast responses. Third we show that by increasing the availability of servers to three combined with SLP convergence for ECDS routing achieved an average 99.4% across all mobility patterns, ranging from 1 to 20 m/s, whilst achieving an average discovery delay rate of 97 ms.

There is an obvious tradeoff between selecting the optimum numbers of servers and applying reliability mechanisms on the underlying transport protocols to achieve high success rates. Generally, the more servers one deploys, the higher the success rates. However, there are situations where deploying relatively high numbers of servers in a network is just not practical or that servers become isolated due to fragmentation. One would also need to consider any overhead and robustness that may be needed in some cases for distributed server or directory synchronization. In these cases, it is clear that SLP convergence can have a significant impact on successful discovery rates. The software being developed allows fine-tuning of such parameters and, therefore, allows creativity for applying such techniques to different environments and settings. Ultimately, such decisions are subject to to the particular dynamics of the network within which the discovery systems is being deployed.

FUTURE WORK

One lesson learned for design feedback into xSLP from these experiments was the problem of responding to multiple queries. Using the various routing algorithms, it is highly likely in a MANET environment to have multiple duplicate delivery of queries, especially with SMF style multicast forwarding. Duplicate delivery also occurs in the conventional Internet but is far less frequent since a mobile topology exacerbates the problem. The server nodes tested within this experiment replied to each query (including duplicates), which caused unnecessary congestion in the network, especially for the Classical flooding case. By modifying this feature of xSLP, we believe we can improve efficiency by adding identifiers to xSLP messages that identify whether a message is a retry or not. In this fashion, we can reply only to nonduplicate requests from the clients by filtering out duplicates from the initial query and additional retry events.

In xSLP we are looking to extend the SLP message format to use XML in order to allow for richer structured advert types, to allow compatibility with other SDPs (e.g., UPnP and Bonjour) and also to integrate more seamlessly for the discovery of higher level services, such as XMPP and Web Services in a mobile setting. In the same way that services are discovered in this report, we are investigating the discovery of multiple brokers to provide bridges between MANET nodes and for bridging between MANET and infrastructure networks for Web services and XMPP. We hope that by providing reliable discovery mechanisms at the MANET level, we can provide reliable cross infrastructure-MANET sessions at the service level that can support a wide range of collaborative applications within a standardized setting.

ACKNOWLEDGMENTS

This work was supported in part by the ONR-sponsored Service-Oriented Networking Over MANET (SONOMA) project. We wish to thank Dr. Santanu Das for his continuing research support. This work was also supported by the OASD (R&E) Networks Program. We would also like to thank Dr. Syed Shah and Dr. Cynthia Dion-Schwarz for their support in sponsoring this research. We wish to thank our many colleagues at NRL and in other technical forums that have helped with related discussions. Finally, we wish to thank the original SLP specification authors for providing a relatively flexible framework for exploring service discovery architectural tradeoffs.

REFERENCES

1. Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations, IETF RFC 2501.
2. JINI: The Jini Web site, see: http:// www.jini.org/.
3. Service Location Protocol, Version 2, http://www.ietf.org/rfc/rfc2608.txt.
4. UPnP, see http://www.upnp.org.
5. Bluetooth SDP, see http://www.bluetooth.org.
6. Bonjour, see http://developer.apple.com/bonjour/.

7. Napster, see http: // www.napster.com/.

8. JXTA, see http://www.jxta.org/.

9. UDDI Spec TC, UDDI Version 3.0.2, September 2004. See: http:// uddi.org/ pubs/ uddi v3.htm.

10. Extensible Messaging and Presence Protocol (XMPP): Core, RFC 3920, see http://www.xmpp.org/.

11. J.P. Macker et al., “Simplified Multicast Forwarding for MANETs,” draft-ietf-manet-smf-08, Work in progress, Nov 08.

12. AgentJ, see http://cs.itd.nrl.navy.mil/work/agentj/index.php.

13. F. Sailhan and V. Issarny, “Scalable Service Discovery for MANET,” Proceedings of the 3rd IEEE Int’l Conf. on Pervasive Computing and Communications (PerCOM), 2005.

14. U. Kozar and L. Tassiulas, “Service Discovery in Mobile Ad hoc Networks: An Overall Perspective on Architectural Choices and Network Layer Support Issues,” Ad Hoc Networks 2, 2004.

15. S. Helal, N. Desai, V. Verma, and C. Lee, “Konark: A Service Discovery and Delivery Protocol for Ad hoc Networks,” Proc. of WCNC, 2003.

16. Protolib, see http://cs.itd.nrl.navy.mil.

17. R. Ogier, “MANET Extension of OSPF Using CDS Flooding,” Proceedings of the 62nd IETF, March 2005. Also see Appendix A of SMF.

18. Optimized Link State Routing (OLSR), IETF RFC 3626.

19. NRLOLSR, see http://cs.itd.nrl.navy.mil.

20. SLP Notification, http://tools.ietf.org/rfc/rfc3082.txt.